Fashion Image Generation using Conditional Deep Convolutional Generative Adversarial Network Based on Text Input

Abstract—Generative Adversarial Network (GAN) is increasingly becoming a popular research area for generating highly realistic data. However one of the pitfalls of GAN is that it cannot be controlled what a conventional GAN will generate. Therefore we extended a Deep Convolutional Generative Adversarial Network (DCGAN) and added conditioning to it, to generate targeted images according to the user input. We used the fashion MNIST dataset to train our generative model. The text given by the user is matched with predefined fashion categories using fuzzy string matching and the model will generate absolutely new fashion images in accordance with the matched fashion category. The inception score of the generated images are analyzed to evaluate the performance of the model.

Index Terms—DNN, Deep Convolutional Network, Generative Adversarial Network

I. INTRODUCTION

Image generation or image synthesis is one of the most common research areas in Machine Learning. Image generation refers to the generation of new images from an existing dataset. During the research, limited availability and quality of training data often act as a bottleneck. No matter how powerful a model is or how many computational resources are available to train it, the model's final performance will suffer if the training data does not accurately represent the data distribution that the model is expected to cover [1]. This is why generative models are becoming increasingly popular in the research field.

To generate any kind of data such as images, text, or video with machine learning, we need to use a generative algorithm. In artificial intelligence (AI), generative modeling is used to develop a depiction of observed phenomena that can be derived from data. Unsupervised machine learning uses this algorithm allowing computers to grasp the real world. Currently, Generative Adversarial Networks (or GANs) are the most promising generative algorithms for image generation.

Generative Adversarial Networks (or GANs) were introduced in 2014 by Ian J. Goodfellow, and co-author, giving machines the gift of imagination. GAN is an unsupervised machine learning (ML) model in which two neural networks compete to improve their prediction accuracy [2]. They've obtained outstanding results in generating examples for Image Datasets, generating realistic photographs, image-to-image synthesis, text-to-image synthesis, and image super-resolution, among other challenges [3].

Despite the fact that GAN models may produce new random believable instances for a given dataset, It has no influence over the visuals that are generated. It can attempt to interpret the intricate correlation between the latent space provided to the generation and the visuals formed. The conditional generative adversarial network (cGAN), is a type of GAN that is capable of generating images conditionally using a generator model. If a class label is available, image generation can be conditional on it, allowing for the targeted generation of images of a specific type [4]. Conditional GANs are used in various fields from creating maps from satellite images [5] to storytelling [6] where Li et al. built a StoryGAN that could generate a sequence of images given text description.

Isola et al. demonstrate GANs, specifically their pix2pix technique, for various image-to-image conversion tasks in their research using Conditional Adversarial Networks [7]. Jun-Yan Zhu presents their popular CycleGAN and a set of highly remarkable image-to-image conversion examples in their 2017 publication titled "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks" [8].

In the text to Image generation field, Han Zhang et al. illustrate the application of GANs. In their 2016 publication, they used their "StackGAN" to generate photographs that are highly realistic from textual information of simple subjects [9]. Moreover, In their 2016 study "Learning What and Where to Draw," Scott Reed et al. extend on this capability in terms of synthesizing images from text according to the recommendations using GANs [10].

In fashion, GANs played an important role by producing creative and unique designs for humans. From research by Kwonsang Sohn, et al. [11], it came out that GAN-generated designs are getting better responses than non-GAN-generated products in the fashion industry. In their 2016 study, Donggeun Yoo et al. illustrate the usage of GANs in generating images of fashion using the photographs of models wearing the item [12]. In a related paper named "ClothingOut: a category-supervised GAN model for clothing segmentation and retrieval", they introduced a novel framework that can automatically produce images of tiled clothing using GAN [13]. Sui et al. introduced FashionGAN using conditional GAN that can generate garment images with an input of a design sketch and the image of the fabric. Liu et al. used multiple discriminators to generate cloth images based on different conditions or requirements [14].

The deep convolutional GAN or DCGAN augmented the

Fig. 1. Overview of the proposed model

Fig. 2. Architecture of conditional GAN

GAN models in terms of configuration and training strategies which brought stability in GAN modeling for a broad spectrum of applications [15]. Makkapati et al. used the power of DCGAN to include symmetry in generated images to make them more realistic [16].

In this paper, we used a conditional Deep Convolutional Generative adversarial Network (DCGAN) to generate fashion images based on user input. The contributions of this paper are as follows:

- Match user text input with cloth category using Levenshtein Distance
- Generate absolutely new cloth image that is visually acceptable according to the text category using DCGAN
- Targeted Image Generation.
- Generation of pre-trained model that can be reused.

II. DATASET

In this paper, we used the Fashion MNIST dataset (https://github.com/zalandoresearch/fashion-mnist) which contains a training set of 60,000 and a test set of 10,000 fashion images. Each image is 784 pixels with a height of 28 pixels and a width of 28 pixels. Both the training and test sets have 785 columns along with a label column. The cloth categories are labelled as: t-shirt/top is 0 , trouser is 1, pullover is 2, dress is 3,coat is 4, sandal is 5, shirt is 6, sneaker is 7,bag is 8 and ankle boot is 9. The images are black and white so an additional channel dimension is added so that it is compatible with the convolutional layers of the proposed model. Then the pixel values are scaled between -1 to 1 as the generator will output in this range.

III. METHODOLOGY

A. Input text processing

The user input text is matched with the pre-defined fashion categories using fuzzy string matching. This will allow slightly differently spelled words to be recognised and matched. We used Levenshtein Distance [17] to find the similarity ratio between two sequences. Lavenshtein distance is a measure of the number of edit operations needed to change one sequence to another.
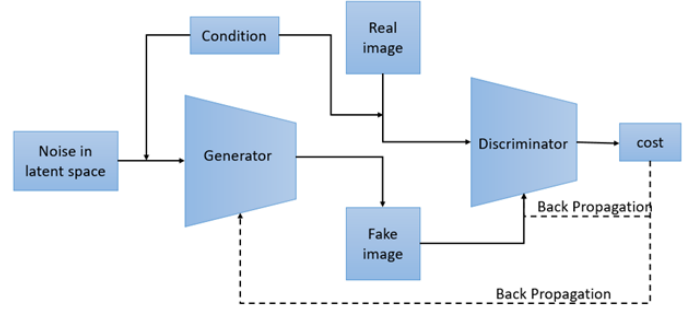
$$
\text{lev}_{x,y}(i,j) = \begin{cases} \max(i,j) \\ \min \begin{cases} \text{lev }_{x,y}(i-1,j)+1 \\ \text{lev }_{x,y}(i,j-1)+1 \\ \text{lev }_{x,y}(i-1,j-1)+1_{(x_i \neq y_j)} \end{cases} \end{cases}
\tag{1}
$$

The above formula (1) is used to measure the Lavenshtein distance between the user input x and predefined cloth category y. The 3 functions correspond to deletion, insertion and substitution function orderly. The Levenshtein similarity ratio can be calculated using formula (2).

$$
\frac{(|x| + |y|) - \text{lev}_{x,y}(i,j)}{|x| + |y|}
\tag{2}
$$

Where $|x|$ and $|y|$ are lengths of the texts. In our paper we used a cutoff value of 70%and the category with the highest similarity ratio is selected.

B. DCGAN architecture

In this section the whole architecture of the conditional DCGAN will be described. We built a DCGAN similar to [15] but added condition to it. It has two components- the generator (G) and the discriminator (D). The generator takes input from the latent space [18], i.e, some random variables using Gaussian Distribution and the label of cloth category. G produces some images *i* according to the label. The discriminator, which is trained with real images along with their labels, classifies the *i* as real or fake. The classification error of D is then used to update both D and G. The overall process is shown in figure [2]

For building the discriminator model D, a convolutional neural network was modelled with LeakyRelu as activation function. The input to the model are 28x28 grayscale images and an integer representing the label of image (cloth category). The label is passed to the D through an embedding layer and reshaped into 28x28 and concatenated with the input images. This creates a two channel input image to be passed on to the convolutional layers. We used 2 by 2 stride for downsampling and stochastic gradient descent (Adam optimizer) with 0.0002 learning rate and a momentum of 0.5 for optimizing.

The generator model G is also a deep convolutional neural network. For input it takes a random variable from the latent space generated by Gaussian Distribution, as well as the label, passed through an embedding layer in a similar way as described for D. G uses LeakyRelu as activation in the fully connected layers and a hyperbolic tangent (tanh) in the output layer. The output is a 28x28 image generated based on the label.

Both G and D are combined to form the condition DCGAN, and then trained. We used a sequential model with an adam optimizer and binary cross entropy for calculating the loss. The training occurs in two phase:

- Phase 1: G creates some images from noise in latent space which are forwarded to D.D is trained on real images. D learns to classify real and fake images. The loss is back propagated and the the weights of D are updated The generator is not updated or trained in this phase.
- Phase 2: G produces a batch of images which are fed to D.D does not have the real images this time. D classifies as real or fake.The loss of D is back propagated to G. The weights of G are updated.But the discriminator weight remains unchanged.

The model is trained with 150 epochs with a batch size of 192. These parameters were selected due to better image generation after trying training the model several times with different hyperparameter tuning.. We used google colab GPU and it took approximately 2.5 hours to train the model. After training we saved the model as a pre-trained model.

C. Loss of condition DCGAN

In the conditional DCGAN model, the generator tries to minimize the loss function in equation (3) and the discriminator tries to maximize it [4], [19].

$$\min_G \max_D Loss = \mathbb{E}_{\boldsymbol{x}}[\log D(\boldsymbol{x} \mid \boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}}[\log(1 - D(G(\boldsymbol{z} \mid \boldsymbol{y})))] \tag{3}$$

Here, y is the label. $D(x|y)$ is the discriminator output for real data x and $D(G(z|y))]$ is the discriminator output for generated fake data by G. Ex represents expected value over real images and Ez represents expected value over all generated fake images $G(z|y)$.

IV. Results

The original images are shown in Figure 3 and the generated images are shown in Figure 4. These images below shows the visual difference of the generated images with the original images, to human eye.

From the given example it can be said that the generated images closely resemble the original image. Even though the images bear close resemblance to the original images, qualitative analysis gives better insight on how the model actually performed. For qualitative analysis of our generated images we used inception score. This was proposed by Salimans, et al. [20] in 2016. It is a popular metric for evaluating images generated by generative models. Inception score is a measure

Fig. 3. Original Fashion MNIST images

Fig. 4. Generated Fashion images

of how realistic the output of a generated model is. As we are generating images of multiple classes, we used the inception model to predict which class the generated image belongs to. The formula for calculating inception score is mentioned in equation (4)

$$\mathrm{I} = \exp\left(\mathbb{E}_{\boldsymbol{x}} \mathrm{KL}(p(y \mid \boldsymbol{x}) \| p(y))\right) \tag{4}$$

Here x is the generated data and y the predicted label by the Inception model. KL is Kullback-Leibler divergence which is basically the conditional probability multiplied by the log of the conditional probability minus the log of the marginal probability.

A pre-trained model was used for the purpose of classification. The model was trained on the training set, to detect the 10 classes of fashion images. Afterwards softmax was used to determine the probability distribution of the predicted classes. 100 images of each class were produced, then the pre-trained model was used to calculate the confidence score of which class the image belongs to. Inception score evaluates the diversity and resemblance of the images to the images the

model was trained on.If both of these characteristics are true, then the score will be high.
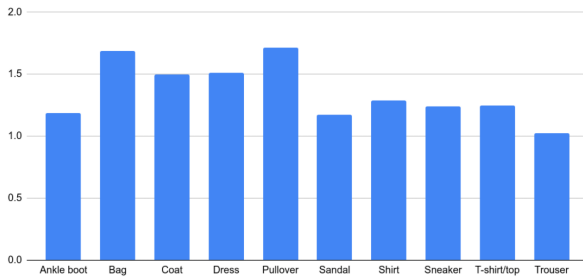

Fig. 5. Inception score for all class labels

From the above bar chart we can see that images of most generated classes scored above 1 in a scale of 2. Which indicates overall acceptable quality of the generated images.

V. FINDINGS

One of the issues we faced with inception score is that it performs well with colored images only. As the images generated by our model are grayscale, the score had high confidence for multiple items, in case of fashion items that look similar to each other. Also for grayscale image output we could not use the Inception V3 as it works only on colored images.
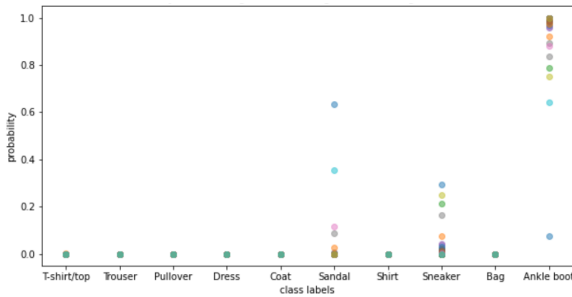

Fig. 6. Probability distribution given that, generated image is Ankle Boot

Above image is the probability distribution for the "Ankle boot" class. We can see, the score of "Ankle boot" class is the highest. Also, similar looking shoe items also had high scores compared to other fashion items. This shows that the classification algorithm found ankle boot is similar to other shoe items, which is a result of poor discriminatory factors.

VI. CONCLUSION

Generative models require very high computation power and a huge amount of time to produce satisfying results. For lack of computing power we had to use grey scale images and images of very small resolution. Even though our model produced acceptable results, it can be improved in various ways. With more computing power and time, it is possible to train a model which is able to generate better looking and higher resolution images. We only used syntactic analysis

of Language to detect the desired class of item to generate. Also, in future, complex language understanding can be used to do contextual analysis and make a model which is able to understand inputs like "Red T-Shirt with long sleeve" and produce respective output. Extensive hyper-parameter tuning can improve the quality of generated images. Finally yet importantly, more complex generative architectures, such as, CYCLE-GAN, ATTENTION-GAN [21], etc can be used to improve the model even further.

REFERENCES

[1] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B. and Bharath, A.A., 2018. Generative adversarial networks: An overview. IEEE Signal Processing Magazine, 35(1), pp.53-65.

[2] I. J. Goodfellow et al., "Generative Adversarial Networks," arXiv.org, 2014. https://arxiv.org/abs/1406.2661.

[3] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-Attention Generative Adversarial Networks," arXiv:1805.08318 [cs, stat], Jun. 2019, [Online]. Available: https://arxiv.org/abs/1805.08318.

[4] M. Mirza and S. Osindero.,2014."Conditional Generative Adversarial Nets," arXiv.org, https://arxiv.org/abs/1411.1784.

[5] Ganguli, S., Garzon, P., Glaser, N., 2019. GeoGAN: A Conditional GAN with Reconstruction and Style Loss to Generate Standard Layer of Maps from Satellite Images. ArXiv, abs/1902.05611.

[6] Li, Y., Gan, Z., Shen, Y., Liu, J., Cheng, Y., Wu, Y., Carin, L., Carlson, D. and Gao, J., 2021. StoryGAN: A Sequential Conditional GAN for Story Visualization. [online] Openaccess.thecvf.com.

[7] P. Isola, J.-Y. Zhu, T. Zhou, and Efros, Alexei A, "Image-to-Image Translation with Conditional Adversarial Networks," arXiv.org, 2016. https://arxiv.org/abs/1611.07004.

[8] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," arXiv.org, 2017. https://arxiv.org/abs/1703.10593.

[9] H. Zhang et al., "StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks," arXiv.org, 2016. https://arxiv.org/abs/1612.03242.

[10] S. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, "Learning What and Where to Draw," arXiv:1610.02454 [cs], Oct. 2016, [Online]. Available: https://arxiv.org/abs/1610.02454.

[11] Sohn, K., Sung, C., Koo, G., Kwon, O., 2020. Artificial intelligence in the fashion industry: consumer responses to generative adversarial network (GAN) technology. International Journal of Retail and Distribution Management, 49, 61-80.

[12] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon, "Pixel-Level Domain Transfer," arXiv:1603.07442 [cs], Nov. 2016, Accessed: Aug. 31, 2021. [Online]. Available: https://arxiv.org/abs/1603.07442.

[13] Zhang, H., Sun, Y., Liu, L., Wang, X., Li, L., Liu, W., 2018. ClothingOut: a category-supervised GAN model for clothing segmentation and retrieval. Neural Computing and Applications, 32, 4519-4530.

[14] Cui, Y.R., Liu, Q., Gao, C., Su, Z., 2018. FashionGAN: Display your fashion design using Conditional Generative Adversarial Nets. Computer Graphics Forum, 37.

[15] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," arXiv.org, 2015. https://arxiv.org/abs/1511.06434.

[16] Makkapati, V. and Patro, A., 2017. Enhancing Symmetry in GAN Generated Fashion Images. Artificial Intelligence XXXIV, pp.405-410.

[17] S. Rani and J. Singh, "Enhancing Levenshtein's Edit Distance Algorithm for Evaluating Document Similarity," Communications in Computer and Information Science, pp. 72–80, 2018, doi: 10.1007/978-981-13-0755-36.

[18] T. P. Van et al., "Interpreting the Latent Space of Generative Adversarial Networks using Supervised Learning," 2020 International Conference on Advanced Computing and Applications (ACOMP), pp. 49–54, Nov. 2020, doi: 10.1109/ACOMP50827.2020.00015.

[19] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs," arXiv.org, 2016. https://arxiv.org/abs/1606.03498.

[20] K. Shmelkov, C. Schmid, and K. Alahari, "How good is my GAN?," arXiv:1807.09499 [cs], Jul. 2018, Accessed: Aug. 31, 2021. [Online]. Available: https://arxiv.org/abs/1807.09499.

[21] X. Chen, C. Xu, X. Yang, and D. Tao, "Attention-GAN for Object Trans-figuration in Wild Images," arXiv:1803.06798 [cs], Mar. 2018, Accessed: Aug. 31, 2021. [Online]. Available: https://arxiv.org/abs/1803.06798.